APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: FAST SEARCH WITH VERY LARGE RESULT SET

APPLICANT: GUENTER RADESTOCK AND OLIVER M. STEINAU

FAST SEARCH WITH VERY LARGE RESULT SET

BACKGROUND

[0001] The following description relates to information management systems and executing a query on a subset of data, for example, to facilitate a fast search with a very large result set.

[0002] An information management system may include a computer system and a data repository. A data repository includes data, such as documents, and may reside on a storage device. In a traditional database system, the data in the data repository typically are referred to as records. Information about the records may be available through an index of the data repository that includes properties, also known as attributes, of the records. In order to retrieve data from the data repository, a user may submit a search query through a computer system. The query may include criteria for searching, such as terms and operators. The information management system may execute the query by reviewing an index of the data repository to find entries in the index that match criteria in the query.

[0003] Depending on criteria specified in a search query and the processes used to execute a search query, the search may require the calculation of "intermediate results." Intermediate results are results that when properly linked together, can be used to generate a set of final results matching the search criteria. For example, a search using the terms "John" and "Smith" with the Boolean operator "AND" placed between the terms may require a first search for "John," which returns a first intermediate result, and a second search for "Smith," which returns a second intermediate result. The intermediate results may be linked together to generate a final result set.

[0004] In some situations only a certain number of results may be desired. Such situations may include a query where only a certain number of results are requested, and/or in calculating intermediate results where one or more of the intermediate results require only a certain number of results. For example, a query may specify that only fifty results meeting the search criteria are requested.

[0005]   In the case of calculating intermediate results, the execution time of a query typically correlates to the size of the intermediate results involved because generating the intermediate results and calculating the required links typically is very time-consuming. The execution of a query is also time-consuming if the results are sorted by an attribute and only a certain number of results are desired. For example, if only fifty results are desired and it is desired that those results are sorted, a query may be executed on all data, all results from that query may be sorted, and then fifty results may be selected.

SUMMARY

[0006]   Described herein are methods and apparatus, including computer program products, that implement techniques for executing a query on a subset of data.

[0007]   In one general aspect, the techniques feature a method of executing a query on a data repository. That method includes receiving a query for execution on data in the data repository; generating an estimate of a number of results of the query; defining a subset of data in the data repository; determining whether to execute the query on the subset of the data; executing the query on the subset of the data to generate a partial set of results if the query is to be executed on the subset of the data, otherwise executing the query on the data repository to generate a complete set of results; and providing query results.

[0008]   Implementations may include one or more of the following features. Providing query results may include making the query results available to an application program. In that case, the method may further include the application program providing query results to a user interface. Determining whether to execute the query on the subset of the data may include determining whether a sufficient number of results will be generated by executing the query on the subset of the data. Determining whether to execute the query on the subset of the data may include estimating whether executing the query on the subset of the data would generate a desired number of results. In that case the method further includes receiving a value representing the desired number of results.

[0009]   The method may further include receiving a value representing the desired number of results. In that case, the query is to be executed on the subset of the data if the

2

estimate of the number of results of the query is greater than a weighted subset estimate

generated in accordance with the following estimation function: $R*\dfrac{N}{stripeSize}*F$, where R

is the number of results desired, N is the total number of possible results, F is an arbitrary

number, and *stripeSize* is the size of the subset of the data; and determining whether to

execute the query on the subset of the data includes generating the weighted subset estimate

and determining whether the estimate of the number of results of the query is greater than the

weighted subset estimate. The method may further include, in response to executing the

query on an (N - 1)th subset of the data, determining whether a sufficient number of results

have been generated; and, if a sufficient number of results have been generated, defining an

Nth subset of the data in the data repository and executing the query on the Nth subset of the

data, otherwise executing the query on the data repository. Generating an estimate of a

number of results of the query may be generated in accordance with the following estimation

functions: $est(NOT)\ =N-est(op)$, $est(AND)=\dfrac{est(op_1)*est(op_2)}{N}$, and

$est(OR)=est(op_1)+est(op_2)-\dfrac{est(op_1)*est(op_2)}{N}$, where *op* is an operand, *est*() signifies an

estimate of the operator or operand in the parenthesis, and N is the total number of possible

results.

[0010]   In another aspect, an information management system includes a data

repository that is configured to store data and one or more processes for executing queries on

the data repository. The one or more processes are operative to receive a query for execution

on data in the data repository; generate an estimate of a number of results of the query; define

a subset of data in the data repository; determine whether to execute the query on the subset

of the data; execute the query on the subset of the data to generate a partial set of results if

the query is to be executed on the subset of the data, otherwise execute the query on the data

repository to generate a complete set of results; and provide query results.

[0011]   Implementations may include one or more of the following features. The

operation of determining whether to execute the query on the subset of the data may include

determining whether a sufficient number of results will be generated by executing the query

on the subset of the data. The operation of providing query results may include making the

query results available to an application program. The operation of determining whether to execute the query on the subset of the data may include estimating whether executing the query on the subset of the data would generate a desired number of results. In that case, the processes may be further operative to receive a value representing the desired number of results. The one or more processes may be further operative to, in response to executing the query on an (N - 1)th subset of the data, determine whether a sufficient number of results have been generated; and define an Nth subset of the data in the data repository and execute the query on the Nth subset of the data if a sufficient number of results have been generated, otherwise execute the query on the data repository.

[0012]   In another aspect, a computer program product is tangibly embodied on an information carrier and the computer program product includes instructions operable to cause data processing apparatus to receive a query for execution on data in a data repository; generate an estimate of a number of results of the query; define a subset of data in the data repository; determine whether to execute the query on the subset of the data; execute the query on the subset of the data to generate a partial set of results if the query is to be executed on the subset of the data, otherwise execute the query on the data repository to generate a complete set of results; and provide query results.

[0013]   Implementations may include one or more of the following features. The operation of providing query results may include making the query results available to an application program. The operation of determining whether to execute the query on the subset of the data may include determining whether a sufficient number of results will be generated by executing the query on the subset of the data. The operation of determining whether to execute the query on the subset of the data may include estimating whether executing the query on the subset of the data would generate a desired number of results. In that case, the computer program product further includes instructions operable to receive a value representing the desired number of results. The computer program product further include instructions operable to, in response to executing the query on an (N - 1)th subset of the data, determine whether a sufficient number of results have been generated; and, if a sufficient number of results have been generated, define an Nth subset of the data in the data

4

repository and execute the query on the Nth subset of the data, otherwise execute the query on the data repository.

[0014]   The methods and apparatus, including computer program products, that implement techniques for executing a query on a subset of data may provide one or more of the following advantages. In scenarios where only a limited number of results or intermediate results are desired, query overhead may be reduced because the query need not be executed on all of the data. The overhead that is reduced may include overhead related to fetching data from a data repository and processor resources related to executing the query. Resources used in relaying the set of results may be reduced due to a smaller result set. Where results are sorted by an attribute, if only a limited number of results are returned from an execution of a query, sorting overhead may be reduced. In order to ensure the overall scheme optimizes resource usage of an information management system, limited portions of data may be searched only when it is estimated that those portions will return a sufficient quantity of results. The reduction of overhead and the optimization of an execution of a query may reduce the amount of time spent executing the query.

[0015]   Details of one or more implementations are set forth in the accompanying drawings and the description below. Other features and advantages may be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0016]   These and other aspects will now be described in detail with reference to the following drawings.

[0017]   FIG. 1 is a flowchart of a method of executing a query.

[0018]   FIG. 2 is a data flow diagram of an information management system.

DETAILED DESCRIPTION

[0019]   The systems and techniques described here relate to information management systems and executing a query on a subset of data.

[0020]   FIG. 1 is a flowchart of a method of executing a query. At 110 the query is received at a computer system of an information management system that includes a data repository. The query may be received via a user interface, such as a graphical user interface of a computer system. The user interface need not be presented by a component of the information management system and may be presented separately from the one or more computer systems in the information management system. For example, a computer program that is a web-based application may forward queries to an information management and display results of those queries. The query includes criteria, which may include both terms and operators.

[0021]   At 120, an estimate of the number of results from an execution of the query on the data repository is generated. The estimate is generated based on the criteria in the query and may be generated using any of a number of techniques. In one technique, where Boolean operators NOT, AND, and OR are supported by the information management system, estimates of the occurrence of terms may be generated in accordance with any technique. Using those estimates, estimation functions for each operator may be used to estimate a total number of results. The following estimation functions may be used:

[0022] $est(NOT) = N - est(op)$

[0023] $est(AND) = \dfrac{est(op_1) * est(op_2)}{N}$

[0024] $est(OR) = est(op_1) + est(op_2) - \dfrac{est(op_1) * est(op_2)}{N}$

[0025]   In those functions, op is an operand; $est()$ signifies an estimate of the operator or operand in the parenthesis; N is the total number of possible results; and, the asterisks represent multiplication. An operand is a term, or a group of terms with operators, that is related to the Boolean operator. For example, in an example query "John AND Smith," John and Smith are both terms and are also operands, whereas AND is an operator. The total number of possible results, N, represents the total number of results that might be returned as a result to a query. For example, if the data repository is composed of documents, each document is a possible result and N represents the total number of documents.

[0026]   An example application of the estimate of a query using the NOT operator might be a query for "NOT Smith." In accordance with the NOT estimation function above, the estimate of the total number of results from the query is the total number of possible results minus the estimate for the number of results of the operand, i.e. the total number of results matching the term Smith.

[0027]   An example application of the estimate of a query using the AND operator might be a query for "John AND Smith." In accordance with the AND estimation function above, the estimate of the total number of results from the query is the total number of results for the first operand, i.e. the total number of results matching John, multiplied by the estimate for the number of results of the second operand, i.e. the total number of results matching Smith, which is divided by the total number of possible results.

[0028]   An example application of the estimate of a query using the OR operator might be a query for "John OR Smith." In accordance with the OR estimation function above, the estimate of the total number of results from the query is the total number of results for the first operand, i.e. the total number of results matching John, added to the estimate for the number of results of the second operand, i.e. the total number of results matching Smith, minus the result of the rest of the estimation function. The rest of the function is the total number of results for the first operand, i.e. the total number of results matching John, multiplied by the estimate for the number of results of the second operand, i.e. the total number of results matching Smith, divided by the total number of possible results.

[0029]   In alternative implementations other techniques and other estimation functions may be used to generate an estimate for the number of results, in addition to or instead of the techniques described above.

[0030]   At 130 a determination is made as to whether the query should be executed on a stripe or on all of the relevant data in the data repository. A stripe is a subset of the relevant data in the data repository, as will be discussed later. If an insufficient number of results are expected or found to be generated based on executing the query on a number of stripes, it may be beneficial to execute the query on a greater portion of the relevant data.

[0031]   The determination at 130 may include determining whether it is worthwhile to execute the query on a single stripe, or any number of stripes. For example, it may be beneficial to sequentially execute the query on all of the stripes, which is still a subset of the relevant data, rather than executing the query on all of the relevant data at once. Any technique may be used to make this determination. For example, an estimate of the total number of results for executing the query on the data repository may be compared against an estimate of the number of results that might be found in a single stripe. The estimate of the total number of results for executing the query on the data repository is the estimate that was calculated in 120. The estimate of the total number of results that might be found in a stripe might be calculated in accordance with the function:

[0032] $$\frac{N}{stripeSize}$$

[0033]   In that function, N is the total number of possible results and *stripeSize* is the size of a stripe. The size of a stripe may be a default value or chosen by other means to optimize the searching of the data repository. In order to compare the estimate of the number of results that might be found in a stripe against the estimate of the number of results that might be found in a search of the data repository, the estimate of the number of results that might be found in a stripe might be multiplied by $R * F$, such that the estimate of 120 is compared against

[0034] $$R * \frac{N}{stripeSize} * F,$$

[0035]   where R is the number of results desired, F is a safety factor, and the asterisks represent multiplication. R may be a number of final results requested. R may be a default value or a value that is input by a user as part of the query. F, the safety factor, is an arbitrary number and may be configured in a user interface. F may be used to compensate for the fact that data might not be evenly distributed in a data repository. F need not be completely arbitrary, and may, for example, be based on tests of different values for F. For example, tests may be run on a sample set of data in a sample data repository using various values for F and it may be determined that four is a desirable value for improving

8

performance. The result of R multiplied by F, multiplied by the estimate for the number of results in a stripe is a weighted stripe estimate and is compared against the number of results estimated for executing the query on the data repository.

[0036]   If the number of results estimated for executing the query on the data repository is greater than the weighted stripe estimate, i.e. if it is likely that there are $R * F$ result documents in a single stripe, a determination may be made to execute the query on a stripe at 150. Otherwise the query is executed on the relevant data at 140.

[0037]   The determination is made at 130 because, although data in the data repository might be distributed evenly such that each stripe should have a proportional number of results, there might not be a sufficiently even distribution to expect enough results to be found in the stripes. In order to offset an uneven distribution, the safety factor F may be set to bias in favor of performing the query on the relevant data set. Thus, for example, if there is a large enough discrepancy between the number of results expected in the repository and a weighted estimate of the number of results expected in the stripes, it may be more efficient to execute the query on the relevant data in the data repository rather than execute the query on the stripes.

[0038]   A stripe is a subset of the data in the data repository. For example, a stripe may contain 50,000 records from a data repository that has 10 million records. In total, all of the stripes that are generated need not cover the entire data repository. For example, in one implementation, each stripe may contain 50,000 records of a data repository that has 10 million records and there may only be a total of 10 stripes. Thus, in that example, the stripes cover a combined 500,000 records of the 10 million records in the data repository. If the results to a query are to be sorted by an attribute, the stripes may be generated from a list of records that are presorted by the attribute (in ascending or descending order). Stripes may be generated in response to each query where the query is executed on the stripes, and all of the stripes may be held in memory such that a performance penalty may occur only for the first execution of the query on a stripe. Although each stripe may come from any section of the data repository, the stripe should not overlap with other stripes and the stripes may be of equivalent size. For example, a first stripe may include records 200-299 in a data repository while a second stripe includes records 300-399 in that data repository.

[0039]   The query is executed on a stripe at 150, such that results to the query match the criteria of the query. The query may be executed using any of a number of techniques and various implementations of the information management system may support any type of format and operators for executing the query. For example, the information management system may support Boolean language searching or natural language searching.

[0040]   At 160 a determination is made as to whether the query should be executed on more stripes. The determination may be made in accordance with any number of techniques. For example, the determination may involve determining whether, based on the number of results generated by execution of the query on stripes so far, there is likely to be enough results in the remaining stripes to satisfy the desired number of results. The determination may allow for uneven distribution of the number of results in each of the stripes. For example, the determination may factor in the possibility that one stripe has a low number of results, which may be compensated by other stripes.

[0041]   One technique for making this determination may include the use of the function:

[0042] $$R * \frac{numberOfEvaluatedStripesSoFar}{totalNumberOfStripes}$$

[0043]   In that function, R is the number of results desired, *totalNumberOfStripes* is the total number of stripes that have been generated based on the data repository, *numberOfEvaluatedStripesSoFar* is the number of stripes on which the query has been executed already, and the asterisk represents multiplication. The result of the function may be compared against the number of results that have been generated by executing the query on stripes. If the number of results that have been generated already is less than the result of the function, a determination may be made that executing the query on one or more remaining stripes is unlikely to generate enough results to meet the desired number of results. Thus, it may be more beneficial to execute the query on the data in the data repository than on the stripes.

[0044]   If the query should not be executed on stripes, the query is executed on the relevant data in the data repository at 140. Otherwise, if the query should continue to be

10

executed on stripes, a determination is made at 170 as to whether there are enough results already generated such that the query need not continue to be executed on stripes. If enough results are already generated, the results are returned at 180. Otherwise, the query is executed on a stripe at 150. The results may be in any format and include any degree of information. For example, in an information management system where records are documents, the results may be a list of documents or the results may be the documents. Each time the query is executed on a stripe, the query is executed on a different stripe. Thus, when results are returned, final results are a combination of the results for each stripe.

[0045]   Although the information management system is discussed in FIG. 1 as including the data repository, in alternative implementations the information management system does not necessarily include the data repository. For example, the information management system may merely access a data repository. The processes of FIG. 1 may be performed on one or more computer systems, and may or might not be performed by the information management system. Although the method of executing a query is shown in FIG. 1 as being composed of several different processes, additional, and/or different processes can be used instead, and all the processes need not be part of a method of executing a query. For example, the query may be executed on stripes without having a determination, after each stripe is searched, as to whether the query should continue to be executed on the stripes. Also, for example, the processes at 110, 120, 130, 140, 150, and 180 may be desirable as a method of executing a query. Similarly, the processes need not be performed in the order depicted. For example, the process at 170 may be performed prior to the process at 160. Also, the processes need not follow the same decision paths. For example, if it is determined that there are not enough results at 170, the next process may be process 130.

[0046]   FIG. 2 is a data flow diagram of an information management system 200. The information management system 200 interacts with a user interface 210 and a data repository 220. The components of FIG. 2 may be part of a single computer system, or they may be part of any number of computer systems. For example, the user interface 210 may be part of the same computer system that has the data repository 220 and that computer system may access a second computer system that includes the information management system 200. The user interface 210 includes any combination of output and input devices. For example,

the user interface 210 may include a display, such as a graphical user interface or a command-line interface, on a display device, in combination with a keyboard and a mouse.

[0047]   The information management system 200 includes a search engine 230, a query optimizer 240, and an engine 250. The search engine 230 manages the execution of queries. The search engine 230 receives queries from the user interface 210 and forwards queries, or parts of queries, to the query optimizer 240 for optimization. In response to forwarding a query, the search engine 230 may receive results from the query optimizer 240. The results may be in any format and include, for example, records, a portion of a record, or a link to a record. In addition, the search engine 230 may manage queries by performing other tasks, such as linking results of intermediate searches for Boolean operators and preparing the results for a computer program that presents the user interface 210.

[0048]   The query optimizer 240 estimates the number of results that should be returned in response to a query, determines whether the query should be executed on a stripe, generates stripes, and executes the query on one or more stripes. For example, in response to receiving a query from the search engine 230, the query optimizer 240 may generate an estimate for the number of results to that query. Based on that number of results, the query optimizer 240 can determine whether the query should be executed on a stripe. If the query optimizer 240 determines to execute the query a stripe, the query optimizer 240 can generate any number of stripes, each of which is subset of the records in the data repository 220. The engine 250 accesses the records, thus the query optimizer 240 can receive portions of the records from the engine 250 as stripes. In alternative implementations the query optimizer 240 may determine whether the query should be executed on any number of stripes. The generation of estimates and determinations by the query optimizer 240 may be performed in accordance with the functions described in relation to FIG. 1.

[0049]   In alternative implementations the information management system need not include or be limited to including the search engine 230, the query optimizer 240, and the engine 250. Also, in alternative implementations the techniques illustrated in FIG. 2 may be performed with more or fewer interactions and in varying order.

[0050]   Although a few implementations have been described in detail above, other modifications are possible. Other implementations may be within the scope of the following claims.